# COMPARATIVE PROPERTIES OF COLLABORATIVE OPTIMIZATION AND OTHER APPROACHES TO MDO

Natalia M. Alexandrov* and Robert Michael Lewis†

* Multidisciplinary Optimization Branch, Mail Stop 159
† Institute for Computer Applications in Science and Engineering, Mail Stop 132C
NASA Langley Research Center, Hampton, Virginia 23681-2199, USA

E-mail: *n.alexandrov@larc.nasa.gov, †buckaroo@icase.edu

## ABSTRACT

We discuss criteria by which one can classify, analyze, and evaluate approaches to solving multidisciplinary design optimization (MDO) problems. Central to our discussion is the often overlooked distinction between questions of formulating MDO problems and solving the resulting computational problem. We illustrate our general remarks by comparing several approaches to MDO that have been proposed.

## INTRODUCTION

There are likely as many definitions of multidisciplinary design optimization (MDO) as there are areas and phases of design. For our discussion, we shall take MDO to mean the *systematic* approach to optimization of complex, coupled engineering systems, where "multidisciplinary" refers to the different aspects that must be included in a design problem. For instance, the design of aircraft involves, among other disciplines, aerodynamics, structural analysis, propulsion, and control. See Sobieszczanski-Sobieski and Haftka (1997), Alexandrov and Hussaini (1997) for overviews of the field.

Broadly speaking, in engineering design problems one attempts to improve or optimize several objectives—frequently competing and conflicting measures of system performance—subject to satisfying a set of design and physical constraints. It is the nature of some of these constraints that distinguishes the engineering design optimization problem from the conventional nonlinear programming problem (NLP). As we discuss, the method of treating the problem constraints provides the defining characteristics for various approaches to solving MDO problems.

The problem solution techniques comprise two major elements: posing the problem as a set of mathematical statements amenable to solution and then defining a procedure for solving the problem once it has been posed. We use the term "formulation" to denote the first element and the term "algorithm" to denote the second. This distinction is crucial, although it is often blurred in presentations of new approaches to MDO. An analysis of an MDO formulation considers such attributes as consistency, well-posedness, equivalence to other formulations, optimality conditions, and sensitivity of solutions to various perturbations. An analysis of an optimization algorithm for solving a given formulation of an MDO problem then considers local convergence rates, global convergence properties, and iteration costs. This work discusses the properties of MDO formulations, including their effect on optimization algorithms.

A sizable body of approaches to solving MDO problems has been proposed over the years. However, there is as yet only limited computational or analytical substantiation of the practical applicability and algorithmic properties of the proposed methods. A number of recent efforts (e.g., Alexandrov and Kodiyalam, 1999) have been aimed at addressing this deficiency. The present work pursues the following objectives:

- The enunciation of a systematic set of criteria for analytical and practical evaluation of MDO methods;
- The classification of MDO formulations according to the approach to maintaining feasibility with respect to analysis and design constraints;
- The analysis of several formulations according to the aforementioned set of criteria in order to give some understanding of the trade-offs among the various formulations;
- A sketch of how features of some specific formulations of MDO problems affect optimization algorithms.

The paper is organized according to this program. We hope to provide the reader some guidance to understanding the algorithmic and performance consequences of choosing one formulation or another for solving an MDO

problem. A more detailed and comprehensive discussion can be found in Alexandrov and Lewis (1999b) .

## CHARACTERISTICS OF MDO FORMULATIONS

By MDO we mean that subset of the total design problem—probably in the conceptual or preliminary phase—that can be formulated as an NLP of the form:

$$\begin{array}{ll} \text{minimize} & f(x, u(x)) \\ \text{subject to} & g(x, u(x)) \geq 0, \end{array} \tag{1}$$

where $x$ is the vector of design variables and $u(x)$ is defined via a block system of equations,

$$A(x, u(x)) = \begin{pmatrix} A_1(x, u_1(x), \ldots, u_N(x)) \\ \vdots \\ A_N(x, u_1(x), \ldots, u_N(x)) \end{pmatrix} = 0, \tag{2}$$

$N$ being the number of blocks. In the context of MDO, the blocks of the system usually represent the state equations for the disciplinary analyses and the necessary interdisciplinary couplings. The state equations normally form a set of coupled differential equations. System (2) is known as the Multidisciplinary Analysis (MDA) system. We have simplified the problem by assuming that the multiple objectives of the system have been synthesized in a single objective $f$, because most extant MDO formulations make this assumption.

At each iteration of a conventional optimization procedure, the design variable vector $x$ is passed to the MDA system. The system is then solved for the state vector $u$. This reduces the dimension of the optimization problem (1) by making it a problem in $x$ only. However, each disciplinary analysis may involve an expensive procedure, say, solving a differential equation. Moreover, to solve the entire MDA system one has to use an iterative procedure that brings the individual analyses into a multidisciplinary equilibrium.

It is the expense of implementing and executing a straightforward, conventional optimization approach to (1) that has mainly motivated researchers to propose alternatives. We now turn to a set of criteria that one can use to evaluate a proposed MDO formulation and gauge the effects of the formulation on optimization algorithms. Some of these considerations, such as disciplinary autonomy and per-iteration cost, are widely noted. However, other criteria seem rarely taken into account, despite their paramount importance.

### Equivalence of Formulations

If we take (1) as representing the MDO problem we ideally wish to solve, it is natural to ask whether an alternative formulation is equivalent to this original problem. There is the question of *mathematical equivalence:* If a vector of design variables solves (1), then, suitably transformed, does it yield a solution of the alternative formulation, and conversely? If the reformulation is not mathematically equivalent, does improvement in the reformulation at least correspond to improvement in the original problem?

Equally important notions of equivalence are more subtle. For instance, there is the question of how optimality conditions, constraint qualifications, and sensitivity calculations in (1) correspond to those in an alternative formulation. These other notions of equivalence have bearing on the practical application of optimization algorithms to the solution of formulations of the MDO problem. A formulation may be mathematically equivalent to (1), yet algorithms applied to its solution may exhibit drastically different behavior than when applied to (1), and may even fail in some cases. We will touch on this point again in our discussion of Collaborative Optimization; further details can be found in Alexandrov and Lewis (1999a) .

### Ease of Implementation

Typically, a tremendous amount of time and effort is required to integrate the analysis software needed for any given formulation of MDO and its solution. In particular, at present there is little MDA capability in existing software, and adding this capability requires a lot of work. This leads to the next consideration.

### Multidisciplinary Analysis

MDA is expensive and requires a considerable effort to implement. The typical approach to avoiding the expense of an explicit MDA is to introduce a relaxation of this system, examples of which we will discuss later. One does requires that the full set of MDA equations be satisfied as one approaches an optimal design.

On the other hand, since the MDA underlies the original problem (1), any attempt to avoid an MDA as an explicit calculation enforced at each step of the optimization must turn some or all of the MDA equations into consistency constraints in the resulting optimization problem. (Provided, of course, that the resulting formulation is equivalent to the original problem (1).) In turn, this means that the optimization algorithm used to solve the reformulation of (1) must shoulder the effort of solving part of the MDA problem. This is problematical if the MDA requires specialized techniques. Moreover, if the interdisciplinary coupling in the MDA has a dominant effect, then avoiding the MDA may be inefficient. Thus, in some cases MDA may be unavoidable.

### Decomposition and Disciplinary Autonomy

This is another very important issue. For many other reasons (e.g., organizational lines of communication, software integration), it is simpler to implement an approach that avoids the iteration required to solve the MDA. One has a natural coarse-grained decomposition along the lines of the disciplines; indeed, the question is not one of decomposition but integration.

Unfortunately, in general one should expect disciplinary autonomy to be in direct conflict with overall computational efficiency in the optimization (see the comments on efficiency below). Nonetheless, where the coupling between disciplines is not too great, disciplinary autonomy might not have too deleterious an effect in this regard.

Some decomposition approaches, such as Collaborative Optimization, make use of disciplinary optimization capabilities, which is seen by some as an attractive feature. However, the ends to which this capability is used is often to solve part of the multidisciplinary analysis problem. Attempting true multiobjective optimization or distributed optimization of a separable objective is not widely done in MDO at present, and is difficult to accomplish for some serious technical reasons.

One other feature of disciplinary autonomy is its inherent parallelism; computation can be carried out independently at the discipline level. However, we do not wish to over-emphasize this as a motivation for decomposition approaches to (1); the simplicity of disciplinary autonomy is its primary attraction. Moreover, the benefits of parallelism are somewhat limited because of the disparity in computational load balancing that often occurs (e.g., computational fluid dynamics takes a much longer time than structural analysis). Often a sequential processing of the disciplines makes more sense for physical and computational reasons.

Work per Iteration vs. Overall Efficiency

An ostensible attraction of approaches to solving (1) that are based on reformulating and solving the problem decomposed along the disciplinary lines is that the cost in each optimization iteration may be much less than that in a single iteration of applying an optimization algorithm directly to the original problem (1). However, if the coupling between any of the disciplines strongly influences the system behavior, this may prove a false economy. As a general rule in optimization, algorithms based on decomposition or separability applied to truly coupled problems are much less efficient, overall, than algorithms that work with the entirely coupled system.

This should not be surprising, as the following simple illustration makes clear. Suppose one wishes to minimize an unconstrained, positive definite quadratic in $x$, and suppose there are $n$ components of $x$. Newton's method costs $O(n^2)$ work for a single iteration, but finds the solution in only one iteration. Steepest descent, on the other hand, costs only $O(n)$ work for a single iteration, but, if the quadratic has highly elongated level sets, then steepest descent will take far more than $n$ iterations to arrive near the solution, negating the smaller per-iteration cost of steepest descent.

This is generally the case for more complicated optimization problems. The Dantzig-Wolfe decomposition for linear programming problems requires less work per iteration, but typically the overall cost of solving problems via this decomposition is greater than that of solving the problems directly via the simplex method, unless the problem exhibits a particular structure. Similar comments hold for solving nonlinear optimization problems.

Dimensionality

Another question that arises is the dimension of the optimization problems that ensue from a given formulation. Arguably, the smaller the dimension, the better. For instance, the MDA (2) can be viewed as a *variable reduction* method insofar as it treats $u$ as a function of the design variables $x$ and thus removes $u$ from the optimization problem.

On the other hand, any attempt to relax the MDA will lead to the introduction of the some of the $u$ into the optimization problem. These additional degrees of freedom in the design problem are then removed by the requirement that the MDA equations be satisfied at the optimal design.

An attractive feature of some decomposition methods is that one can also eliminate some of the design variables $x$ from the system-level optimization problem. This can be done, for instance, if the effect of some of the design variables is restricted to a specific discipline. This strategy is followed in Collaborative Optimization, as we will discuss.

Another question related to dimensionality is that of the bandwidth and strength of the interdisciplinary coupling. Depending on how (1) is formulated and solved, the amount of information that must be exchanged between disciplines, and the frequency with which information must be exchanged, may vary markedly. This, in turn, is related to the extent to which the problem is being treated in a decomposed way, which is itself related to the efficiency with which the overall problem will be solved.

Treatment of Feasibility

This consideration, central to the taxonomy of MDO methods, forms the subject of the next section.

Robustness

Given the expense of design optimization, one should demand robustness from any proposed formulation and algorithm for its solution. Some approaches to solving the MDO problem (1) actually amount to solving some manner of relaxation or approximation of (1), but may encounter difficulties as one makes the relaxation more like the original problem. We give an example of this below, where a proposed formulation suffers from a deficiency that can defeat numerical optimization algorithms. One can fine-tune this approach so that for a given problem it stably produces answers (though answers only to a relaxed version of the design problem); the necessity of fine-tuning might be acceptable for some situations, but

not in others. More generally, one would prefer to have a robust approach from the start.

Solubility by Available Algorithms / Convergence

One question that is, surprisingly, sometimes overlooked, is the existence of optimization algorithms that will solve a particular formulation of the MDO problem. It is possible to reformulate the MDO problem in a way that is difficult to solve reliably. We touch on this in an example below; in Alexandrov and Lewis (1999a) one can find a detailed illustration of two equivalent formulations that manifest drastically different behavior when conventional optimizers are applied to their solution. In particular, formulations that lead to nonconvex bilevel and multilevel problems are hard to solve reliably, and are expensive to solve, as well.

We must also ask about the convergence properties of optimization algorithms applied to a given formulation of the MDO problem. At the very least, one would ask for a guarantee of convergence from an arbitrary starting design to at least a local optimizer of the design problem, since such guarantees are typically part of the analysis of modern nonlinear optimization algorithms. There is also the question of the rate at which optimization algorithms will converge, which in part determines overall efficiency of the optimization.

The availability of algorithms to solve a particular formulation of the MDO problem may limit the set of problems for which the formulation is useful. Similarly, proposed algorithms for the solution of a given approach will be limited by their convergence properties.

## CLASSIFICATION OF MDO FORMULATIONS

We now turn to a classification of MDO formulations. The taxonomy we propose differs from other schemes that have appeared (e.g., Cramer, *et al.*, 1994, Balling and Sobieski, 1994) .

The proposed classification is based on the way that a formulation handles the constraints explicit and implicit in (1). These constraints comprise the following:

- *Disciplinary analysis constraints*, which are equality constraints implicit in disciplinary analyses;
- *Design constraints*, which are general nonlinear constraints, some at the disciplinary level, others that couple outputs from different disciplines;
- *Interdisciplinary consistency constraints*, which are auxiliary constraints introduced to relax interdisciplinary coupling.

We will illustrate these distinctions for the following two-discipline instance of the MDO problem (1):

$$\begin{array}{ll} \text{minimize} & f(x_0, R_1(u_1(x)), R_2(u_2(x))) \\ \text{subject to} & g_0(x_0, S_1(u_1(x)), S_2(u_2(x))) \geq 0 \\ & g_1(x_0, x_1, u_1(x)) \geq 0 \\ & g_2(x_0, x_2, u_2(x)) \geq 0, \end{array} \tag{3}$$

where, given $x$, $(u_1, u_2)$ is the solution of the MDA

$$A_1(x_0, x_1, u_1, T_1(u_2)) = 0 \tag{4}$$
$$A_2(x_0, x_2, u_2, T_2(u_1)) = 0. \tag{5}$$

The design variables $x$ have been partitioned into $x = (x_0, x_1, x_2)$. The system-level design variables $x_0$ are shared by both disciplines. The disciplinary design variables $x_1$ and $x_2$ are specific to disciplines 1 and 2.

The operators $R_i$ and $S_i$ indicate that perhaps only a subset of the state variables $u_i$ is required to evaluate the system-level objective $f$ and the design constraint $g_0$. The constraints $g_1, g_2$ are the disciplinary design constraints.

The operators $T_i$ indicate that the output of one disciplinary analysis may need to be transformed before being passed to the other discipline. Equations (4)–(5) are the disciplinary analysis constraints. They distinguish the design problem from the conventional NLP. At this stage, there are no explicit interdisciplinary consistency constraints.

Alternative formulations of (3) rely on the introduction of auxiliary variables and consistency constraints. For instance, we can rewrite the MDA (4)–(5) as

$$A_1(x_0, x_1, u_1, u_{12}) = 0 \tag{6}$$
$$A_2(x_0, x_2, u_2, u_{21}) = 0 \tag{7}$$
$$u_{12} - T_1(u_2) = 0 \tag{8}$$
$$u_{21} - T_2(u_1) = 0. \tag{9}$$

Thus we can rewrite (3) as an equivalent problem in $(x_0, x_1, x_2, u_{12}, u_{21})$:

$$\begin{array}{ll} \text{minimize} & f(x_0, R_1(u_1(x, u_{12})), R_2(u_2(x, u_{21}))) \\ \text{subject to} & g_0(x_0, S_1(u_1(x_0, x_1, u_{12})), \\ & \qquad S_2(u_2(x_0, x_2, u_{21}))) \geq 0 \\ & g_1(x_0, x_1, u_1(x_0, x_1, u_{12})) \geq 0 \\ & g_2(x_0, x_2, u_2(x_0, x_2, u_{21})) \geq 0 \\ & u_{12} - T_1(u_2(x_0, x_2, u_{12})) = 0 \\ & u_{21} - T_2(u_1(x_0, x_1, u_{21})) = 0, \end{array} \tag{10}$$

where, given $(x, u_{12}, u_{21})$, $u_1$ and $u_2$ are found by solving the disciplinary analysis equations

$$A_1(x_0, x_1, u_1, u_{12}) = 0$$
$$A_2(x_0, x_2, u_2, u_{21}) = 0.$$

Equations (8) and (9) are examples of interdisciplinary consistency constraints. The degrees of freedom introduced by expanding the set of optimization variables to include $u_{12}, u_{21}$ are removed by the consistency constraints.

Approaches to MDO problems are generally based on techniques for eliminating variables from transformations of the original problem. The variables are eliminated by enforcing various subsets of the constraints in

different ways. We will say that an MDO formulation is *closed* with respect to a given set of constraints if the formulation—rather than an optimization algorithm for its solution—assumes that these constraints are satisfied at every iteration of the optimization. If the formulation does not necessarily assume that a set of constraints is satisfied, we will say that that formulation is *open* with respect to the set of constraints.

For instance, consider conventional optimization applied to the formulation (3). We perform a multidisciplinary analysis at each step. This corresponds to maintaining closure of all the disciplinary analysis and interdisciplinary consistency constraints in (10).

More generally, MDO formulations are characterized by the constraint sets with respect to which they are open or closed. In addition, a particular optimization method applied to the formulation may enforce closure with respect to additional sets of constraints. However, we again stress the importance of differentiating between the properties of a formulation and the properties of an algorithm for its solution.

We distinguish the following classes of formulations, based on their treatment of constraints. We refer the reader to Alexandrov and Lewis (1999b) for details.

- CDA/OD/CIC: Closed disciplinary analysis, open design constraints, closed interdisciplinary consistency constraints. This is the conventional formulation (3), also known as the Multidisciplinary Feasible (MDF) formulation in Cramer *et al.* (1994). Further closure with respect to disciplinary and system-level design constraints is determined by the kind of optimization algorithm used. Another example of a formulation included in this large class is Bi-Level Integrated System Synthesis (BLISS) by Sobieszczanski-Sobieski *et al.* (1998).

- CDA/CD/OIC: Closed disciplinary analysis, closed design constraints, open interdisciplinary consistency constraints. Examples of this class include Collaborative Optimization (Braun, 1996, Braun *et al.*, 1997) and the formulation proposed in Walsh *et al.* (1992) .

- CDA/OD/OIC: Closed disciplinary analysis, open design constraints, open interdisciplinary consistency constraints. The Individual Discipline Feasible (IDF) approaches discussed in Cramer *et al.* (1994) and Lewis (1997) are examples of this class. Again, closure with respect to disciplinary and system-level design constraints is determined by the kind of optimization algorithm used.

- OA/OD/OIC: Open analysis, open design constraints, open interdisciplinary consistency constraints. Simultaneous Analysis and Design (Haftka, *et al.*, 1990) is an example of this class of formulation. Once again, closure with respect to disciplinary and system-level design constraints is determined by the kind of optimization algorithm used.

## A STUDY OF TWO FORMULATIONS

We now consider members of two classes of formulations and comment on some of their features in terms of our previous discussion. We choose the two representatives because their approaches are very similar, but a seemingly slight difference in the problem statements causes their analytical behavior and the effect on optimization algorithms to differ significantly. For simplicity of exposition, the discussion will proceed in terms of the two-discipline problem of the preceding section.

### Collaborative Optimization

In our classification scheme, Collaborative Optimization (CO) (Braun, 1996, Braun *et al.*, 1997) is closed with respect to disciplinary analyses, closed with respect to design constraints, and open with respect to interdisciplinary consistency constraints. CO has three salient features.

First, CO is a nonconvex, nonlinear bilevel optimization problem of a special structure.

Second, the only constraints of the system-level problem are the interdisciplinary consistency constraints that are designed to drive the discrepancy among the disciplinary inputs and outputs to zero. The values of the system-level constraints are computed by solving disciplinary optimization problems. The number of the consistency constraints is related to the number of the disciplines, the number of variables shared among the disciplines, and the number of outputs exchanged among the disciplines. The form of the consistency constraints characterizes different instances of CO.

Finally, the disciplinary problems are NLP whose objective is to minimize the discrepancy between the system-level variables and their local, disciplinary copies, subject to satisfying the design constraints. The disciplinary constraints do not depend explicitly on the system-level variables that are passed down to the disciplinary problems as parameters.

Reformulating (10) along the lines of CO, we introduce new disciplinary design variables $x_0^1, x_0^2$ that serve to further relax the coupling between the disciplines through the shared system-level design variables $x_0$. At the system level, we introduce new variables $w_1, w_2$ and $y_1, y_2$ to relax the coupling between disciplines through the system-level objective $f$ and constraint $g_0$, respectively. The system-level constraint $g_0$ will be treated as an additional discipline. The resulting system-level NLP

in $(x_0, u_{12}, u_{21}, w_1, w_2, y_1, y_2)$ is

$$\begin{array}{ll} \min & f(x_0, R_1(u_1(x_0, x_{12}, y_1)), R_2(u_2(x_0, x_{21}, y_2))) \\ \text{s.t.} & c_0(x_0, y_1, y_2, x_0^0(x_0, y_1, y_2), \\ & \quad z_1(x_0, y_1, y_2), z_2(x_0, y_1, y_2)) = 0, \\ & c_1(x_0, u_{12}, y_1, x_0^1(x_0, x_{12}, y_1), \\ & \quad x_1(x_0, x_{12}, y_1), u_1(x_0, x_{12}, y_1)) = 0 \\ & c_2(x_0, u_{21}, y_2, x_0^2(x_0, x_{21}, y_2), \\ & \quad x_2(x_0, x_{21}, y_2), u_2(x_0, x_{21}, y_2) = 0 \end{array} \tag{11}$$

where the $c_i$ are the consistency constraints we will shortly describe. We compute $x_0^1(x_0, x_{12}, y_1)$, $x_1(x_0, x_{12}, y_1)$, and $u_1(x_0, x_{12}, y_1)$ by solving the following minimization problem in $(x_0^1, x_1)$ at the discipline level:

$$\begin{array}{ll} \text{minimize} & \| x_0^1 - x_0 \|^2 \\ & + \| S_1(u_1(x_0^1, x_1)) - y_1 \|^2 \\ & + \| T_1(u_1(x_0^1, x_1)) - u_{12} \|^2 \\ \text{subject to} & g_1(x_0^1, x_1, u(x_0^1, x_1)) \geq 0. \end{array} \tag{12}$$

An analogous problem for discipline 2 defines $x_0^2(x_0, x_{21}, y_2)$, $x_2(x_0, x_{21}, y_2)$, and $u_2(x_0, x_{21}, y_2)$:

$$\begin{array}{ll} \text{minimize} & \| x_0^2 - x_0 \|^2 \\ & + \| S_2(u_2(x_0^2, x_2)) - y_2 \|^2 \\ & + \| T_2(u_2(x_0^2, x_2)) - u_{21} \|^2 \\ \text{subject to} & g_2(x_0^2, x_2, u(x_0^2, x_2)) \geq 0. \end{array} \tag{13}$$

In the disciplinary problems, $u_1$ and $u_2$ are computed via the disciplinary analyses

$$\begin{aligned} A_1(x_0^0, x_1, u_1, u_{12}) &= 0 \\ A_2(x_0^1, x_2, u_2, u_{21}) &= 0. \end{aligned}$$

"Discipline 0" is introduced to treat the system-level design constraints; the associated disciplinary problem in $(x_0^0, z_1, z_2)$ is

$$\begin{array}{ll} \text{minimize} & \| x_0^0 - x_0 \|^2 + \| z_1 - y_1 \|^2 \\ & \quad + \| z_2 - y_2 \|^2 \\ \text{subject to} & g_0(x_0^0, z_1, z_2) \geq 0. \end{array} \tag{14}$$

The introduction of disciplinary minimization subproblems of this form is the distinctive characteristic of CO. The subproblems are independent of one another and can be solved autonomously at the discipline level. In doing so, the disciplinary design variables $x_i$ and the disciplinary state variables $u_i$ are eliminated from the system-level problem.

Information from the solutions of the disciplinary problems (12)–(14) is then used to define the system-level consistency constraints $c_i$. Here we will discuss two definitions of $c_i$.

The first instance of CO we discuss is the form in which CO is usually presented (Balling and Wilkinson (1997), Braun (1996), Braun *et al.* (1997), Sobieski and Kroo (1996)) . In this formulation, the consistency condition is to drive to zero the minimum value of subproblems (12)–(14). The system-level consistency constraints are simply the optimal values of the objectives in (12)–(14). Bearing in mind that

$$\begin{aligned} x_0^0 &= x_0^0(x_0, y_1, y_2) \\ x_0^1 &= x_0^1(x_0, x_{12}, y_1) \\ x_0^2 &= x_0^2(x_0, x_{21}, y_2) \\ u_1 &= u_1(x_0^1(x_0, x_{12}, y_1), x_1(x_0, x_{12}, y_1)) \\ u_2 &= u_2(x_0^2(x_0, x_{21}, y_2), x_1(x_0, x_{21}, y_2)) \\ z_1 &= z_1(x_0, y_1, y_2) \\ z_2 &= z_2(x_0, y_1, y_2), \end{aligned} \tag{15}$$

we have the consistency constraints

$$\begin{aligned} &c_0(x_0, y_1, y_2) = \\ &\quad \| x_0^0 - x_0 \|^2 + \| z_1 - y_1 \|^2 + \| z_2 - y_2 \|^2 \\ &c_1(x_0, x_{12}, y_1) = \| x_0^1 - x_0 \|^2 \\ &\quad + \| S_1(u_1) - y_1 \|^2 + \| T_1(u_1) - u_{12} \|^2 \\ &c_2(x_0, x_{21}, y_2) = \| x_0^2 - x_0 \|^2 \\ &\quad + \| S_2(u_2) - y_2 \|^2 + \| T_2(u_2) - u_{21} \|^2 . \end{aligned} \tag{16}$$

We call this version $CO_2$, where the subscript 2 refers to the fact that the $c_i$ are sums of squares.

An alternative to the system-level consistency condition (16) is to explicitly match the system-level variables with their subsystem counterparts computed in subproblems (12)–(14):

$$\begin{aligned} &c_0(x_0, y_1, y_2) = (x_0^0 - x_0, z_1 - y_1, z_2 - y_2) \\ &c_1(x_0, u_{12}, y_1) = \\ &\quad (x_0^1 - x_0, S_1(u_1) - y_1, T_1(u_1) - u_{12}) \\ &c_2(x_0, u_{21}, y_2) = \\ &\quad (x_0^2 - x_0, S_1(u_2) - y_2, T_1(u_2) - u_{21}), \end{aligned} \tag{17}$$

again, keeping in mind (15). We will denote this approach as $CO_1$. In general, this leads to more system-level equality constraints than does $CO_2$. The latter usually reduces this vector of information about inconsistency into as many constraints as there are subsystems (Braun, 1996), but the vector may be reduced to a single scalar.

Note that the number of system-level variables in the system-level problem (11) depends on the number of shared variables $x_0$ and the bandwidth of the interdisciplinary coupling, as manifest in $u_{12}, u_{21}, w_1, w_2, y_1, y_2$. The introduction of these auxiliary variables suggest that CO will be best suited for problems with a narrow coupling bandwidth. CO possesses a marked degree of disciplinary autonomy. However, as computational experience (Alexandrov and Kodiyalam, 1998, Kodiyalam, 1998) and analysis (Alexandrov and Lewis, 1999a) reveal, the approach has a number of intrinsic analytical and computational difficulties.

For instance, in $CO_2$, one can show that Lagrange multipliers *never* exist for the system-level constraints

(16). This means that nonlinear optimization algorithms will fail if one attempts to truly enforce the consistency conditions $c_i = 0$; instead, one must be content with a relaxation $\| c_i \| \leq \varepsilon$ for some suitably small $\varepsilon$. In practice, the larger $\varepsilon$ at the system level and the tighter the convergence criteria for the subsystems, the better are the chances that an optimization algorithm applied to $CO_2$ will find a solution.

This problem does not occur in $CO_1$. However, in order to compute the first derivatives in the system-level optimization problem we must compute *second* derivatives at the disciplinary level; this is a consequence of the fact that CO involves a bilevel optimization problem, in which the system-level optimization problem involves the output of the disciplinary level optimization problems.

Other difficulties in both $CO_1$ and $CO_2$ derive from the fact that CO leads to a nonlinear bilevel optimization problem. In particular, the constraints (16) and (17) possess features that can cause distress for standard optimization algorithms. These include the potential for nonsmoothness due to multiple local minima in the disciplinary problems (12)–(14). For further details, see Alexandrov and Lewis (1999a).

Individual Discipline Feasible Approaches

The IDF formulation provides another way to avoid the expensive MDA iteration. It is closed with respect to disciplinary analyses, open with respect to design constraints, and open with respect to interdisciplinary consistency constraints. Further closure with respect to design constraints or system level constraints is determined by the kind of optimization algorithm used.

Various forms of IDF were discussed in Cramer *et al.* (1994) and Lewis (1997). The term "Individual Discipline Feasible" originally referred to maintaining closure with respect to the disciplinary analysis constraints at each optimization iteration, but not closure with respect to multidisciplinary analysis coupling until a solution is reached. However, the question of disciplinary design constraints was not really treated in earlier discussion of IDF. Here we discuss the original IDF approach and its elaboration that treats the design constraints explicitly.

We begin with (10). The IDF formulation discussed in (Cramer *et al.*, 1994, Lewis, 1997) is closed with respect to the disciplinary analysis constraints:

$$\begin{array}{ll} \text{minimize} & f(x_0, R_1(u_1), R_2(u_2)) \\ \text{subject to} & g_0(x_0, S_1(u_1), S_2(u_2)) \geq 0 \\ & g_1(x_0, x_1, u_1) \geq 0 \\ & g_2(x_0, x_2, u_2) \geq 0 \\ & u_{12} - T_1(u_2) = 0 \\ & u_{21} - T_2(u_1) = 0, \end{array} \tag{18}$$

where $u_1 = u_1(x_0, x_1, u_{12})$ and $u_2 = u_2(x_0, x_2, u_{21})$ are required only to satisfy the disciplinary analysis constraints; i.e., $u_1$ and $u_2$ are defined by solving

$$\begin{aligned} A_1(x_0, x_1, u_1, u_{12}) &= 0 \\ A_2(x_0, x_2, u_2, u_{21}) &= 0. \end{aligned}$$

In contrast to CO, the IDF formulation is a single-level NLP.

In the IDF approach, further closure with respect to disciplinary design constraints or system level constraints is determined by the kind of optimization algorithm used. Ideally, one would be able to start with design variables $(x_0, x_1, x_2)$ for which the disciplinary design constraints defined by the $g_i$ are satisfied. One could then apply an optimization algorithm that maintained feasibility with respect to these constraints so that all subsequent designs obtained in the course of the optimization satisfied the disciplinary design constraints, thereby accomplishing the same end that CO achieves through the definition of its disciplinary optimization problems.

On the other hand, one might rightly object that it will, in general, be difficult to find initial design variables $(x_0, x_1, x_2)$ for which the disciplinary design constraints are satisfied. To address this problem, we can expand the space along the lines of CO as follows:

$$\begin{array}{ll} \text{minimize} & f(x_0, R_1(u_1), R_2(u_2)) \\ \text{subject to} & g_0(x_0^0, y_1, y_2) \geq 0 \\ & g_1(x_0^1, x_1, u_1) \geq 0 \\ & g_2(x_0^2, x_2, u_2) \geq 0 \\ & u_{12} - T_1(u_2) = 0 \\ & u_{21} - T_2(u_1) = 0 \\ & x_0 - x_0^0 = 0 \\ & x_0 - x_0^1 = 0 \\ & x_0 - x_0^2 = 0 \\ & y_1 - S_1(u_1) = 0 \\ & y_2 - S_2(u_2) = 0. \end{array} \tag{19}$$

This relaxes the requirement that the disciplinary design constraints be satisfied with the system-level values of $x_0$. In particular, we now have the flexibility to select the initial $x_0^i$ and $y_i$ in a way that the disciplinary design constraints are satisfied, exactly as in CO. One can then apply an optimization algorithm that enforces feasibility with respect to the disciplinary design constraints.

It is straightforward to verify that IDF is equivalent to the original MDO. This makes IDF easy to analyze; for instance, if standard constraint qualifications are satisfied by the original problem, then they also hold for the IDF formulation. The convergence properties of optimization algorithms applied to IDF are those of the algorithms applied to conventional NLP. Given a good solver for equality constrained optimization problems, the method is expected to be efficient.

Similarly to CO, IDF is intended for problems with small bandwidth of interdisciplinary coupling, and the

problem of decomposition is similar to that of CO. Also similarly to CO, formulations that arise from IDF have more optimization variables that those arising from MDF.

Importantly, although IDF maintains autonomy with respect to analyses, it lacks CO's autonomy with respect to disciplinary optimization. That is, while the analyses are performed autonomously during the analysis stage, the coupling is restored during the optimization step computation. This brings back the difficulties of integration. On the other hand, as previously noted, the disciplinary optimization in CO is actually part of addressing the MDA, not actually improving the disciplinary objectives of the true design problem, such as weight or the lift-to-drag ratio.

## CONCLUDING REMARKS

This paper has introduced a portion of an extensive effort aimed at furthering the understanding of the analytical and computational properties of methods for solving MDO problems and at proposing efficient methods based on this understanding. We emphasized the distinction between an MDO formulation and an optimization algorithm and discussed a new, comprehensive classification of MDO formulations based on the way the constraint sets are explicitly treated in a formulation. Members of two formulation classes were discussed as an example.

When considering an MDO problem statement, both the problem formulation and the available nonlinear programming algorithms for solving the formulation must be examined carefully with the following questions in mind: How is the new formulation related to the basic NLP formulation of the MDO problem? Does the new formulation lead to an optimization problem that is not amenable to solution by existing optimization algorithms? For instance, is the new formulation a nonconvex multilevel optimization problem?

The motivation for a prospective formulation must be continually re-examined. It may be discovered that what one attempts to accomplish via a formulation with a difficult structure is more easily accomplished by a judicious choice of an optimization algorithm.

As a rule, full disciplinary autonomy with respect to optimization is in direct conflict with computational efficiency for general problems. If the coupling between the disciplinary subsystems is sufficiently broad and strong, one may wish to consider formulations that sacrifice some degree of autonomy for the sake of efficiency, especially since there are optimization algorithms to which such problems are amenable.

## REFERENCES

Alexandrov, N. M. and Hussaini, M. Y., Eds. (1997), *Multidisciplinary Design Optimization: State of the Art*, SIAM, Philadelphia.

Alexandrov, N. M. and Kodiyalam, S. (1998), "Initial results of an MDO method evaluation study", *AIAA Paper 98-4884*.

Alexandrov, N. M. and Lewis, R. M. (1999a), "Analytical and computational aspects of collaborative optimization", to be submitted to *Engineering Optimization*.

Alexandrov, N. M. and Lewis, R. M. (1999b), "Problem formulation and algorithms in multidisciplinary optimization", ICASE report. In preparation.

Balling, R. J. and Sobieszczanski-Sobieski S. (1994), "Optimization of coupled systems: a critical overview of approaches". *AIAA Paper 94-4330*.

Balling, R. J. and Wilkinson, C. A. (1997), "Execution of multidisciplinary design optimization approaches on common test problems", *AIAA J.*, 35, pp. 178–186.

Braun, R. (1996), *Collaborative Optimization: An architecture for large-scale distributed design*, PhD thesis, Stanford University.

Braun, R. D., Moore, A. A. and Kroo, I. M. (1997), "Collaborative approach to launch vehicle design", *J. of Spacecraft and Rockets*, 34, pp. 478–486.

Cramer, E., Dennis, Jr., J. E., Frank, P., Lewis, R. and Shubin, G. (1994), "Problem formulation for multidisciplinary design optimization", *SIAM J. on Optimization*, 4, pp. 754–776.

Haftka, R. T., Gürdal, Z. and Kamat, M. P. (1990), *Elements of Structural Optimization*, Kluwer Academic Publishers, Dordrecht.

Lewis, R. M. (1997), "Practical Aspects of Variable Reduction Formulations and Reduced Basis Algorithms in Multidisciplinary Design Optimization", in *Multidisciplinary Design Optimization: State-of-the-Art*, Alexandrov, N. M. and Hussaini, M. Y., Eds., SIAM.

Sobieski, I. and Kroo, I. (1996), "Aircraft design using collaborative optimization". *AIAA Paper 96-0715*.

Sobieszczanski-Sobieski, J., Agte, J. and Sandusky, Jr., R. (1998), "Bi-level integrated system synthesis (BLISS)", *NASA/TM-1998-208715*.

Sobieszczanski-Sobieski, J. and Haftka, R. T. (1997), "Multidisciplinary aerospace design optimization: survey of recent developments", *Structural Optimization*, 14, pp. 1–23.

Walsh, J., Young, K., Pritchard, J., Adelman, H. and Mantay, W. (1994), "Multilevel decomposition approach to integrated aerodynamic/dynamic/structural optimization of helicopter rotor blades", *NASA/TM-1994-109084*.